Available online at www.sciencedirect.com
SCIENCE DIRECT®

Appl. Comput. Harmon. Anal. 19 (2005) 359–385

Applied and Computational Harmonic Analysis

www.elsevier.com/locate/acha

Shadow block iteration for solving linear systems obtained from wavelet transforms

Yao Lu [a,1], Lixin Shen [b,*], Yuesheng Xu [c,d,2]

[a] *Academy of Mathematics and System Sciences, Academia Sinica, Beijing, 100080, PR China*
[b] *Department of Mathematics, Western Michigan University, Kalamazoo, MI 49008, USA*
[c] *Department of Mathematics, Syracuse University, Syracuse, NY 13244, USA*
[d] *Institute of Mathematics, Academy of Mathematics and System Science, Chinese Academy of Science, Beijing, 100080, PR China*

Received 25 October 2004; revised 15 January 2005; accepted 31 January 2005

Available online 8 August 2005

Communicated by Charles K. Chui

Abstract

Matrices resulting from wavelet transforms have a special "shadow" block structure, that is, their small upper left blocks contain their lower frequency information. Numerical solutions of linear systems with such matrices require special care. We propose shadow block iterative methods for solving linear systems of this type. Convergence analysis for these algorithms are presented. We apply the algorithms to three applications: linear systems arising in the classical regularization with a single parameter for the signal de-blurring problem, multilevel regularization with multiple parameters for the same problem and the Galerkin method of solving differential equations. We also demonstrate the efficiency of these algorithms by numerical examples in these applications.

* Corresponding author.
E-mail addresses: luy@amss.ac.cn (Y. Lu), lixin.shen@wmich.edu (L. Shen), yxu06@syr.edu (Y. Xu).

[1] Supported in part by the Natural Science Foundation of China under grant 10371122, by Guangxi Provincial Natural Science Foundation of China under grant 0339036 and by Professor Yuesheng Xu's grant from the program of "One Hundred Distinguished Young Chinese Scientists" of the Chinese Academy of Sciences.

[2] Supported in part by the US National Science Foundation under grants CCR-0407476, by the Natural Science Foundation of China under grant 10371122 and by the Chinese Academy of Sciences under the program of "One Hundred Distinguished Young Chinese Scientists."

1063-5203/$ – see front matter
doi:10.1016/j.acha.2005.01.004

Keywords: Wavelets; Block iterations; Regularization; Deblurring

1. Introduction

Numerous applications in sciences and engineering, as well as in computational, applied and pure mathematics give rise to problems of solving linear equations

$$A\mathbf{u} = \mathbf{g}, \tag{1}$$

where A is an $N \times N$ invertible matrix, $\mathbf{g} \in \mathbb{R}^N$ is a given vector, and $\mathbf{u} \in \mathbb{R}^N$ denotes the unknown vector. There are two well-known types of methods to solve the linear equation. One type is the direct methods such as Gaussian elimination and LU factorization method. Methods of this type compute the exact solution $\mathbf{u}$ after finite steps. Another type is the iterative methods such as the Jacobi method and Gauss–Seidel method, which generally produce a good approximation of the exact solution $\mathbf{u}$. The literature for both two methods is vast, the interested readers can consult the books, for instance, [1] and [8]. For linear equations involving matrices with Toeplitz, Hankel structures, along with other patterns of structure, many fast and efficient algorithms have been developed by exploiting the special structure of such matrices.

Wavelet analysis has played an important role in a broad range of applications, including signal processing, data and image compression, solutions of partial differential equations, modeling multiscale phenomena, and statistics [6,12]. There seem to be no limits to the subjects where it may have utility. The use of wavelet methods leads to linear systems with a coefficient matrix having the special structure. Often, after using wavelet transforms the small upper left block of the resulting matrix is a *shadow* of the original matrix in the sense that it contains its lower frequency information. To visually see this point, we consider the "Barbara" image (Fig. 1(a)) as the matrix A in (1). The first level and the third level wavelet decomposition of A are shown in Fig. 1(b) and (c) where the main information of A are concentrated on the upper left blocks (shadows). This suggests that we should take advantage of such a new structure to

(a) (b) (c)

Fig. 1. (a) Original "Barbara" image; (b) The first level wavelet decomposition; (c) The third level wavelet decomposition.

introduce novel block iterative algorithms in solving linear system (1). We next describe the main idea. Instead of solving (1) directly, we solve the following linear system

$$PAQ^T\tilde{\mathbf{u}} = \tilde{\mathbf{g}}, \tag{2}$$

where P and Q are two invertible transform matrices related to the wavelet we used, $\tilde{\mathbf{g}} = P\mathbf{g}$. Figure 5(a) is a plot of the matrix A and Fig. 5(b) is a plot of the transformed matrix PAQ^T of A. Clearly, most of largest amplitudes of PAQ^T concentrate on a small subblock. This motivates us to develop a block iterative algorithm, which tailors to the structure of PAQ^T, to solve the linear system (2). The final solution is simply $\mathbf{u} = Q^T\tilde{\mathbf{u}}$. The efficiency of the algorithm is illustrated by several examples. As we will see in the next section, the proposed approach is a novel hybrid of a direct method with an iterative algorithm.

We organize this paper into five sections. In Section 2, we describe four algorithms based on different matrix splitting methods. We provide in Section 3 sufficient conditions for convergence of these algorithms and prove special results on convergence of the algorithms for symmetric positive definite matrices. In Section 4, we present three applications of these algorithms to illustrate the use of these algorithms. The first two applications are for the signal de-blurring problem and the third application is the numerical solution of boundary value problems. Numerical examples are presented to demonstrate the performance of these algorithms. In Section 5, we make concluding remarks.

2. Description of algorithms

In this section, we describe the shadow block iteration methods. For notational convenience, we use A for an $N \times N$ matrix obtained from wavelet transforms. In other words, the matrix A has the special structure illustrated in Section 4. Specifically, we assume that A has the block form

$$A = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 + A_5 \end{bmatrix}, \tag{3}$$

where A_1 is $N_1 \times N_1$ matrix and A_4, A_5 are $N_2 \times N_2$ matrices with $N = N_1 + N_2$, and they satisfy the properties that

- A_1 has a small size with respect to the size of A and the inverse of A_1 is easy to compute (often, it can be computed by a direct method);
- A_4 is an invertible matrix and its inverse A_4^{-1} is easy to compute (it can also be computed by a direct method).

We develop shadow block iteration methods based on splitting methods of the matrix A. For this purpose, we recall the usual matrix splitting method. There exists a large group of iterative algorithms for solving the linear equation (1). One class of iterative methods including Jacobi and Gauss–Seidel iterations is based on a splitting (B, C) of A, i.e.,

$$A = B - C, \tag{4}$$

where B is nonsingular and the inverse B^{-1} is easy to compute. In particular, we can express the matrix A as the matrix sum $A = L + D + U$, where D is the diagonal matrix with same diagonal elements of A

and L and U are strictly lower and upper triangular matrices, respectively. The Jacobi iterative scheme is associated with $B = D$ and $C = -(L + U)$ while the Gauss–Seidel iterative scheme is associated with $B = (L + D)$ and $C = -U$. Using the splitting (4), starting from $\mathbf{u}_0 = \mathbf{0}$, an approximate solution $\mathbf{u}_m$ is computed by

$$B\mathbf{u}_{m+1} = C\mathbf{u}_m + \mathbf{g}. \tag{5}$$

It is well known (see, for example, [17]) that the iterative method (5) converges to the unique solution $\mathbf{u} = A^{-1}\mathbf{g}$ for any initial guess $\mathbf{u}_0$ if and only if the spectral radius of the matrix $B^{-1}C$ is strictly less than 1. Clearly, the effectiveness of the iterative scheme depends on how to split A into B and C. The smaller the spectral radius of the matrix $B^{-1}C$ is, the faster the algorithm converges.

We now return to the description of our method. Based on the structure of matrix A described earlier we propose three different ways to split A into B and C. Specifically, we have three types of splitting.

Type 1:

$$B = \begin{bmatrix} A_1 & 0 \\ 0 & A_4 \end{bmatrix}, \qquad C = -\begin{bmatrix} 0 & A_2 \\ A_3 & A_5 \end{bmatrix}. \tag{6}$$

Type 2:

$$B = \begin{bmatrix} A_1 & A_2 \\ 0 & A_4 \end{bmatrix}, \qquad C = -\begin{bmatrix} 0 & 0 \\ A_3 & A_5 \end{bmatrix}. \tag{7}$$

Type 3:

$$B = \begin{bmatrix} A_1 & 0 \\ A_3 & A_4 \end{bmatrix}, \qquad C = -\begin{bmatrix} 0 & A_2 \\ 0 & A_5 \end{bmatrix}. \tag{8}$$

The key idea of these splitting methods is that matrix B reflects *lower* frequency of A and matrix C mainly contains *higher* frequency of A and that the inverse of B is easy to compute. Note that the proposed three types of splitting are different from traditional Jacobi and Gauss–Seidel splitting.

The proposed splitting schemes suggest us three different iterative algorithms to solve the linear equation (5). In the algorithms to be described, we choose $\mathbf{u}_0$ to be any initial guess and write

$$\mathbf{g} = \begin{bmatrix} \mathbf{g}^L \\ \mathbf{g}^H \end{bmatrix}, \qquad \mathbf{u}_m = \begin{bmatrix} \mathbf{u}_m^L \\ \mathbf{u}_m^H \end{bmatrix}$$

for any m, where $\mathbf{g}^L$, $\mathbf{u}_m^L \in \mathbb{R}^{N_1}$ and $\mathbf{g}^H$, $\mathbf{u}_m^H \in \mathbb{R}^{N_2}$. They correspond to lower and higher frequency of $\mathbf{g}$ and $\mathbf{u}_m$, respectively.

Corresponding to the splitting (6), we have the following algorithm.

Algorithm 1. *Iterate*

$$\mathbf{u}_{m+1}^L = A_1^{-1}\big(\mathbf{g}^L - A_2\mathbf{u}_m^H\big),$$
$$\mathbf{u}_{m+1}^H = A_4^{-1}\big(\mathbf{g}^H - [\,A_3 \quad A_5\,]\,\mathbf{u}_m\big)$$

for $m = 0, 1, \ldots,$ *until it converges.*

Corresponding to the splitting (7), we have the following algorithm.

Algorithm 2. *Iterate*

$$\mathbf{u}_{m+1}^H = A_4^{-1}\big(\mathbf{g}^H - [\,A_3 \quad A_5\,]\,\mathbf{u}_m\big),$$
$$\mathbf{u}_{m+1}^L = A_1^{-1}\big(\mathbf{g}^L - A_2\mathbf{u}_{m+1}^H\big)$$

for $m = 0, 1, \ldots,$ *until it converges.*

Corresponding to the splitting (8), we have the following algorithm.

Algorithm 3. *Iterate*

$$\mathbf{u}_{m+1}^L = A_1^{-1}\big(\mathbf{g}^L - A_2\mathbf{u}_m^H\big),$$
$$\mathbf{u}_{m+1}^H = A_4^{-1}\big(\mathbf{g}^H - A_3\mathbf{u}_{m+1}^L - A_5\mathbf{u}_m^H\big)$$

for $m = 0, 1, \ldots,$ *until it converges.*

The difference among (6), (7), and (8) is now evident from the implementations in Algorithms 1–3. In Algorithm 1, both $\mathbf{u}_{m+1}^L$ and $\mathbf{u}_{m+1}^H$ can be parallel computed. In Algorithm 2, one first computes $\mathbf{u}_{m+1}^H$ and then computes $\mathbf{u}_{m+1}^L$ with updated information $\mathbf{u}_{m+1}^H$ while in Algorithm 3, one first computes $\mathbf{u}_{m+1}^L$ and then computes $\mathbf{u}_{m+1}^H$ with updated information $\mathbf{u}_{m+1}^L$. We further remark that we can precalculate inverses of A_1 and A_4 as they do not change in the iterations.

Algorithm 4 is a combination of Algorithms 2 and 3.

Algorithm 4. *Iterate*

$$\mathbf{u}_{m+1/2}^H = A_4^{-1}\big(\mathbf{g}^H - [\,A_3 \quad A_5\,]\,\mathbf{u}_m\big),$$
$$\mathbf{u}_{m+1}^L = A_1^{-1}\big(\mathbf{g}^L - A_2\mathbf{u}_{m+1/2}^H\big),$$
$$\mathbf{u}_{m+1}^H = A_4^{-1}\big(\mathbf{g}^H - A_3\mathbf{u}_{m+1}^L - A_5\mathbf{u}_{m+1/2}^H\big)$$

for $m = 0, 1, \ldots,$ *until it converges.*

All of these algorithms are hybrid of direct methods and iterative methods. They based on efficient direct methods for finding A_1^{-1} and A_4^{-1}. We will address this point through examples in Section 4.

3. Convergence analysis

The main purpose of this section is to provide convergence results for the algorithms introduced in the last section. To this end, we introduce some notations. For any vector $\mathbf{x} \in \mathbb{R}^N$,

$$\|\mathbf{x}\| = \left(\sum_{i=1}^{N} |x_i|^2\right)^{1/2}$$

is the Euclidean norm of $\mathbf{x}$. For any matrix A, we define its matrix norm by

$$\|A\| = \sup_{\mathbf{x}\neq 0} \frac{\|A\mathbf{x}\|}{\|\mathbf{x}\|}.$$

For a square matrix $A \in \mathbb{R}^{N\times N}$, the spectral radius of the matrix A is $\rho(A) = \max|\lambda_i|$, where λ_i are eigenvalues of A.

Our first proposition regards convergence of Algorithm 1.

Proposition 1. *Suppose that*

$$\max\{\|A_4^{-1}A_3\|^2, \|A_1^{-1}A_2\|^2 + \|A_4^{-1}A_5\|^2\} + \|A_4^{-1}A_3\|\|A_4^{-1}A_5\| < 1. \tag{9}$$

Then, Algorithm 1 *converges for any initial guess* $\mathbf{u}_0$.

Proof. Using splitting (6), a direct computation leads to matrix

$$B^{-1}C = -\begin{bmatrix} 0 & A_1^{-1}A_2 \\ A_4^{-1}A_3 & A_4^{-1}A_5 \end{bmatrix}.$$

Correspondingly, for $\mathbf{x} = \left[\begin{smallmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{smallmatrix}\right]$, we obtain that

$$B^{-1}C\mathbf{x} = -\begin{bmatrix} A_1^{-1}A_2\mathbf{x}_2 \\ A_4^{-1}A_3\mathbf{x}_1 + A_4^{-1}A_5\mathbf{x}_2 \end{bmatrix}$$

and thus,

$$\begin{aligned}\|B^{-1}C\mathbf{x}\|^2 &= \|A_1^{-1}A_2\mathbf{x}_2\|^2 + \|A_4^{-1}A_3\mathbf{x}_1 + A_4^{-1}A_5\mathbf{x}_2\|^2 \\ &\leqslant \|A_4^{-1}A_3\|^2\|\mathbf{x}_1\|^2 + \big(\|A_1^{-1}A_2\|^2 + \|A_4^{-1}A_5\|^2\big)\|\mathbf{x}_2\|^2 \\ &\quad + 2\|A_4^{-1}A_3\|\|A_4^{-1}A_5\|\|\mathbf{x}_1\|\|\mathbf{x}_2\|.\end{aligned}$$

We assume that $\|\mathbf{x}\| = 1$. Since

$$\|\mathbf{x}_1\|^2 + \|\mathbf{x}_2\|^2 = \|\mathbf{x}\|^2 = 1 \quad \text{and} \quad 2\|\mathbf{x}_1\|\|\mathbf{x}_2\| \leqslant 1, \tag{10}$$

we have that

$$\begin{aligned}\|B^{-1}C\|^2 &= \sup\{\|B^{-1}C\mathbf{x}\|^2 :\ \|\mathbf{x}\| = 1\} \\ &\leqslant \max\{\|A_4^{-1}A_3\|^2, \|A_1^{-1}A_2\|^2 + \|A_4^{-1}A_5\|^2\} + \|A_4^{-1}A_3\|\|A_4^{-1}A_5\|.\end{aligned}$$

This together with (9) leads to $\rho(B^{-1}C) < 1$ because $\rho(B^{-1}C) \leqslant \|B^{-1}C\| < 1$. □

The following proposition is about convergence of Algorithm 2.

Proposition 2. *Suppose that*

$$\big(\|A_1^{-1}A_2A_4^{-1}\|^2 + \|A_4^{-1}\|^2\big)\big(\max\{\|A_3\|^2, \|A_5\|^2\} + \|A_3\|\|A_5\|\big) < 1. \tag{11}$$

Then, Algorithm 2 *converges for any initial guess* $\mathbf{u}_0$.

Proof. Using splitting (7), we have that

$$B^{-1}C = \begin{bmatrix} A_1^{-1}A_2A_4^{-1}A_3 & A_1^{-1}A_2A_4^{-1}A_5 \\ -A_4^{-1}A_3 & -A_4^{-1}A_5 \end{bmatrix}.$$

As in the proof of Proposition 1, for $\mathbf{x} = \left[\begin{smallmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{smallmatrix}\right]$, we obtain that

$$B^{-1}C\mathbf{x} = \begin{bmatrix} A_1^{-1}A_2A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2) \\ -A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2) \end{bmatrix}$$

and thus,

$$\begin{aligned}\big\|B^{-1}C\mathbf{x}\big\|^2 &= \big\|A_1^{-1}A_2A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2)\big\|^2 + \big\|A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2)\big\|^2 \\ &\leqslant \big(\big\|A_1^{-1}A_2A_4^{-1}\big\|^2 + \big\|A_4^{-1}\big\|^2\big)\|A_3\mathbf{x}_1 + A_5\mathbf{x}_2\|^2.\end{aligned}$$

Again, for any vector $\mathbf{x}$ with $\|\mathbf{x}\| = 1$, we have that

$$\begin{aligned}\|A_3\mathbf{x}_1 + A_5\mathbf{x}_2\|^2 &\leqslant \big(\|A_3\|\|\mathbf{x}_1\| + \|A_5\|\|\mathbf{x}_2\|\big)^2 \\ &= \|A_3\|^2\|\mathbf{x}_1\|^2 + 2\|A_3\|\|A_5\|\|\mathbf{x}_1\|\|\mathbf{x}_2\| + \|A_5\|^2\|\mathbf{x}_2\|^2.\end{aligned}$$

Hence, by (10),

$$\|A_3\mathbf{x}_1 + A_5\mathbf{x}_2\|^2 \leqslant \max\big\{\|A_3\|^2, \|A_5\|^2\big\} + \|A_3\|\|A_5\|. \tag{12}$$

By definition,

$$\begin{aligned}\big\|B^{-1}C\big\|^2 &= \sup\big\{\big\|B^{-1}C\mathbf{x}\big\|^2\colon \|\mathbf{x}\| = 1\big\} \\ &\leqslant \big(\big\|A_1^{-1}A_2A_4^{-1}\big\|^2 + \big\|A_4^{-1}\big\|^2\big)\big(\max\big\{\|A_3\|^2, \|A_5\|^2\big\} + \|A_3\|\|A_5\|\big).\end{aligned}$$

This together with (11) leads to $\rho(B^{-1}C) < 1$ because $\rho(B^{-1}C) \leqslant \|B^{-1}C\| < 1$. □

The next proposition concerns convergence of Algorithm 3.

Proposition 3. *Suppose that*

$$\big\|A_1^{-1}A_2\big\|^2 + \big\|A_4^{-1}\big(A_3A_1^{-1}A_2 - A_5\big)\big\|^2 < 1. \tag{13}$$

Then, Algorithm 3 *converges for any initial guess* $\mathbf{u}_0$.

Proof. Using splitting (8), we conclude that

$$B^{-1}C = \begin{bmatrix} 0 & -A_1^{-1}A_2 \\ 0 & A_4^{-1}(A_3A_1^{-1}A_2 - A_5) \end{bmatrix}.$$

For $\mathbf{x} = \left[\begin{smallmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{smallmatrix}\right]$, we have that

$$B^{-1}C\mathbf{x} = \begin{bmatrix} -A_1^{-1}A_2\mathbf{x}_2 \\ A_4^{-1}(A_3A_1^{-1}A_2 - A_5)\mathbf{x}_2 \end{bmatrix}$$

and thus, for $\|\mathbf{x}\| = 1$,

$$\begin{aligned}\big\|B^{-1}C\mathbf{x}\big\|^2 &= \big\|A_1^{-1}A_2\mathbf{x}_2\big\|^2 + \big\|A_4^{-1}\big(A_3A_1^{-1}A_2 - A_5\big)\mathbf{x}_2\big\|^2 \\ &\leqslant \big\|A_1^{-1}A_2\big\|^2 + \big\|A_4^{-1}\big(A_3A_1^{-1}A_2 - A_5\big)\big\|^2.\end{aligned}$$

By definition,

$$\big\|B^{-1}C\big\|^2 \leqslant \big\|A_1^{-1}A_2\big\|^2 + \big\|A_4^{-1}\big(A_3A_1^{-1}A_2 - A_5\big)\big\|^2.$$

This together with (13) leads to $\rho(B^{-1}C) < 1$ because $\rho(B^{-1}C) \leqslant \|B^{-1}C\| < 1$. □

The next proposition in this section is about convergence of Algorithm 4.

Proposition 4. *Suppose that*

$$\left(\left\|A_1^{-1}A_2A_4^{-1}\right\|^2 + \left\|A_4^{-1}\left(A_3A_1^{-1}A_2 - A_5\right)A_4^{-1}\right\|^2\right)\left(\max\left\{\|A_3\|^2, \|A_5\|^2\right\} + \|A_3\|\|A_5\|\right) < 1. \tag{14}$$

Then, Algorithm 4 *converges for any initial guess* $\mathbf{u}_0$.

Proof. As pointed out before, Algorithm 4 is a combination of Algorithms 2 and 3. Algorithm 4 can be rewritten as $\mathbf{u}_{m+1} = F\mathbf{u}_m + G\mathbf{g}$ with

$$F = \begin{bmatrix} 0 & -A_1^{-1}A_2 \\ 0 & A_4^{-1}(A_3A_1^{-1}A_2 - A_5) \end{bmatrix} \begin{bmatrix} A_1^{-1}A_2A_4^{-1}A_3 & A_1^{-1}A_2A_4^{-1}A_5 \\ -A_4^{-1}A_3 & -A_4^{-1}A_5 \end{bmatrix}$$

$$= \begin{bmatrix} A_1^{-1}A_2A_4^{-1}A_3 & A_1^{-1}A_2A_4^{-1}A_5 \\ -A_4^{-1}(A_3A_1^{-1}A_2 - A_5)A_4^{-1}A_3 & -A_4^{-1}(A_3A_1^{-1}A_2 - A_5)A_4^{-1}A_5 \end{bmatrix}$$

and

$$G = \begin{bmatrix} 0 & -A_1^{-1}A_2 \\ 0 & A_4^{-1}(A_3A_1^{-1}A_2 - A_5) \end{bmatrix} \begin{bmatrix} A_1^{-1} & -A_1^{-1}A_2A_4^{-1} \\ 0 & A_4^{-1} \end{bmatrix} + I.$$

For $\mathbf{x} = \left[\begin{smallmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{smallmatrix}\right]$, we have that

$$F\mathbf{x} = \begin{bmatrix} A_1^{-1}A_2A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2) \\ -A_4^{-1}(A_3A_1^{-1}A_2 - A_5)A_4^{-1}(A_3\mathbf{x}_1 + A_5\mathbf{x}_2) \end{bmatrix}$$

and

$$\|F\mathbf{x}\|^2 \leqslant \left(\left\|A_1^{-1}A_2A_4^{-1}\right\|^2 + \left\|A_4^{-1}\left(A_3A_1^{-1}A_2 - A_5\right)A_4^{-1}\right\|^2\right)\|A_3\mathbf{x}_1 + A_5\mathbf{x}_2\|^2.$$

We further assume $\|\mathbf{x}\| = 1$. Using (12) and (14), we obtain that

$$\|F\|^2 = \sup\left\{\|F\mathbf{x}\|^2 \colon \|\mathbf{x}\| = 1\right\} < 1.$$

This leads to $\rho(F) < 1$. Hence, Algorithm 4 converges for any initial guess $\mathbf{u}_0$. □

We next present special results on convergence of our proposed algorithms when the matrix A is symmetric and positive definite. We begin with the following lemma.

Lemma 1. *Let* A *be a real symmetric and positive definite matrix. Suppose that* A *can be written as* $A = B - C$ *with* B *being nonsingular and* $B^T + C$ *being positive definite, then* $\rho(B^{-1}C) < 1$.

Proof. Let λ be any eigenvalue of $B^{-1}C$. Then, there exists a nonzero vector $\mathbf{x}$ such that

$$B^{-1}C\mathbf{x} = \lambda\mathbf{x},$$

that is,

$$C\mathbf{x} = \lambda B\mathbf{x}.$$

This leads to equations that

$$B(1-\lambda)\mathbf{x} = B\mathbf{x} - C\mathbf{x} = A\mathbf{x}$$

and

$$C(1-\lambda)\mathbf{x} = \lambda B\mathbf{x} - \lambda C\mathbf{x} = \lambda A\mathbf{x}.$$

We observe that $\lambda \neq 1$, since otherwise $A\mathbf{x} = (B-C)\mathbf{x} = \mathbf{0}$ implies $\mathbf{x} = \mathbf{0}$. On the other hand, we have that

$$\begin{aligned}
|1-\lambda|^2\mathbf{x}^*\big(B^T + C\big)\mathbf{x} &= (1-\lambda)^*\mathbf{x}^*\big(B^T + C\big)(1-\lambda)\mathbf{x} \\
&= (1-\lambda)^*\mathbf{x}^*B^T(1-\lambda)\mathbf{x} + (1-\lambda)^*\mathbf{x}^*C(1-\lambda)\mathbf{x} \\
&= (A\mathbf{x})^*(1-\lambda)\mathbf{x} + (1-\lambda)^*\mathbf{x}^*\lambda A\mathbf{x} \\
&= \mathbf{x}^*A^*\mathbf{x} - \lambda^*\lambda\mathbf{x}^*A\mathbf{x} \\
&= \big(1-|\lambda|^2\big)\mathbf{x}^*A\mathbf{x},
\end{aligned}$$

where we use c^* for the conjugate of a complex number c and $\mathbf{x}^*$ for the conjugate transpose of a vector $\mathbf{x}$.

Because of the positive definiteness of matrices A and $B^T + C$, the left-hand side of above the equation is positive and $\mathbf{x}^*A\mathbf{x} > 0$. Thus, $1 - |\lambda|^2 > 0$, i.e., $|\lambda| < 1$. Therefore, $\rho(B^{-1}C) < 1$. □

A direct consequence of applying Lemma 1 to Algorithm 1 leads to the following result.

Proposition 5. *Let A be a symmetric and positive definite with a splitting as* (6)*. If matrix*

$$\begin{bmatrix} A_1 & -A_2 \\ -A_3 & A_4 - A_5 \end{bmatrix}$$

is positive definite, then $\rho(B^{-1}C) < 1$ and Algorithm 1 *is convergent.*

Note that when A is symmetric, the convergence analysis for Algorithm 2 is similar to that for Algorithm 3. We consider only Algorithm 3.

Proposition 6. *Let A be a symmetric and positive definite with a splitting as* (8)*. If $A_4 - A_5$ is positive definite, then $\rho(B^{-1}C) < 1$ and Algorithm* 3 *is convergent.*

Proof. From splitting (8), we have that

$$B^T + C = \begin{bmatrix} A_1 & 0 \\ 0 & A_4 - A_5 \end{bmatrix}.$$

Because both A_1 and $A_4 - A_5$ are positive definite, matrix $B^T + C$ is also positive definite. Therefore, Lemma 1 gives $\rho(B^{-1}C) < 1$ and thus, Algorithm 3 is convergent. □

Proposition 7. *Let A be a symmetric and positive definite with partition* (3)*. If A_4 and A_5 are symmetric and $A_4 - A_5$ is positive definite, then Algorithm* 4 *is convergent.*

Proof. Let

$$D_1 = \begin{bmatrix} A_1 & 0 \\ 0 & A_4 \end{bmatrix}, \quad D_2 = -\begin{bmatrix} 0 & 0 \\ 0 & A_5 \end{bmatrix}, \quad L = -\begin{bmatrix} 0 & 0 \\ A_3 & 0 \end{bmatrix}, \quad U = -\begin{bmatrix} 0 & A_2 \\ 0 & 0 \end{bmatrix}.$$

We rewrite Algorithm 4 as

$$(D_1 - U)\mathbf{u}_{m+\frac{1}{2}} = (L + D_2)\mathbf{u}_m + \mathbf{g},$$
$$(D_1 - L)\mathbf{u}_{m+1} = (U + D_2)\mathbf{u}_{m+\frac{1}{2}} + \mathbf{g}.$$

Combining the above equations gives

$$\begin{aligned}(D_1 - L)\mathbf{u}_{m+1} &= (U + D_2)(D_1 - U)^{-1}(L + D_2)\mathbf{u}_m + \big((U + D_2)(D_1 - U)^{-1} + I\big)\mathbf{g} \\ &= (U + D_2)(D_1 - U)^{-1}(L + D_2)\mathbf{u}_m + (D_1 + D_2)(D_1 - U)^{-1}\mathbf{g}.\end{aligned}$$

By introducing two matrices

$$B = (D_1 - U)(D_1 + D_2)^{-1}(D_1 - L)$$

and

$$C = (D_1 - U)(D_1 + D_2)^{-1}(U + D_2)(D_1 - U)^{-1}(D_2 + L),$$

we write Algorithm 4 in the matrix form

$$B\mathbf{u}_{m+1} = C\mathbf{u}_m + \mathbf{g}.$$

To show that this algorithm converges, it suffices to prove that $B^T + C$ is positive definite. To this end, we note that

$$(U + D_2)(D_1 - U)^{-1} = -I + (D_2 + D_1)(D_1 - U)^{-1}.$$

Hence, we have that

$$C = (U - D_1)(D_1 + D_2)^{-1}(D_2 + L) + (D_2 + L) = (D_2 + U)(D_1 + D_2)^{-1}(D_2 + L)$$

and it follows that

$$B^T + C = (D_1 - L)^T(D_1 + D_2)^{-1}(D_1 - L) + (D_2 + L)^T(D_1 + D_2)^{-1}(D_2 + L). \tag{15}$$

Since both A and $A_4 - A_5$ are positive definite, then A_1, A_4 and

$$D_1 + D_2 = \begin{bmatrix} A_1 & \\ & A_4 - A_5 \end{bmatrix}$$

are positive definite. Noting that $\det(D_1 - L) = \det(D_1) = \det(A_1)\det(A_4) \neq 0$, i.e., the matrix $D_1 - L$ is non-singular, the first term in the right-hand side of Eq. (15) is positive definite and the second term is positive semi-definite. Therefore, $B^T + C$ is positive definite. By applying Lemma 1, Algorithm 4 is convergent. □

Next, we present a sufficient and necessary condition for the positive definiteness of matrix $A_4 - A_5$.

Proposition 8. *Let A be a symmetric and positive definite with a partition* (3). *Suppose that A_4 and A_5 are symmetric, then $A_4 - A_5$ is positive definite if and only if*

$$\rho\big(A_4^{-1}A_5\big) < 1. \tag{16}$$

Proof. Using Lemma 1 with $B := A_4$ and $C := -A_5$, we conclude that if $A_4 - A_5$ is positive definite, condition (16) holds.

Conversely, we assume that condition (16) holds. We then observe that $I - A_4^{-1}A_5$ is nonsingular and thus, $A_4 - A_5$ is nonsingular. Assume that $A_4 - A_5$ were not positive definite. This implies that there would exist a nonzero vector $\mathbf{x}_0$ such that

$$\mathbf{x}_0^T(A_4 - A_5)\mathbf{x}_0 < 0.$$

We define a vector sequence $\mathbf{x}_k$, $k = 1, 2, \ldots$, recursively by

$$\mathbf{x}_k = A_4^{-1}A_5\mathbf{x}_{k-1}.$$

Clearly, we have that

$$\mathbf{x}_k = \left(A_4^{-1}A_5\right)^k\mathbf{x}_0, \quad k = 1, 2, \ldots.$$

By hypothesis (16), we find that $\lim_{k\to\infty} \mathbf{x}_k = \mathbf{0}$ and consequently,

$$\lim_{k\to\infty} \mathbf{x}_k^T(A_4 - A_5)\mathbf{x}_k = 0. \tag{17}$$

By symmetry, we have that $A_4^T = A_4$, $A_5^T = A_5$. Moreover, by the recursive formula, $A_4\mathbf{x}_k = A_5\mathbf{x}_{k-1}$. Using these identities, we conclude that

$$\mathbf{x}_{k-1}^T(A_4 - A_5)\mathbf{x}_{k-1} - \mathbf{x}_k^T(A_4 - A_5)\mathbf{x}_k = \left(\mathbf{x}_{k-1}^T - \mathbf{x}_k^T\right)(A_4 + A_5)(\mathbf{x}_{k-1} - \mathbf{x}_k).$$

The positive definiteness of A implies the positive definiteness of $A_4 + A_5$, which ensures that the right-hand side of the above equation is positive for $\mathbf{x}_{k-1} \neq \mathbf{x}_k$. Thus, we obtain that

$$\mathbf{x}_k^T(A_4 - A_5)\mathbf{x}_k \leqslant \mathbf{x}_{k-1}^T(A_4 - A_5)\mathbf{x}_{k-1} \leqslant \cdots \leqslant \mathbf{x}_0^T(A_4 - A_5)\mathbf{x}_0 < 0$$

for all $k = 0, 1, \ldots$. This contradicts (17) and the contradiction shows that matrix $A_4 - A_5$ is positive definite. □

Combining Propositions 7 and 8, we have the following corollary.

Corollary 1. *Let A be a symmetric and positive definite with a partition* (3)*. If A_4 and A_5 are symmetric and $\rho(A_4^{-1}A_5) < 1$, then Algorithm* 4 *converges.*

Roughly speaking, the first four propositions indicate that the desirable structure of A would be that the absolute values of the entries of A_2, A_3, and A_5 are small enough in comparison to those of A_1 and A_4. However, an arbitrary invertible matrix A may not have such structure. The idea is to find suitable matrices P and Q so that PAQ^T has the desired structure. This idea is motivated by wavelet transforms but not restricted to them. Using this idea, instead of solving (1), we solve the linear system (2). The key issue then becomes to find appropriate matrices P and Q for a given matrix A such that once the matrix PAQ^T is split into one of the form in (6), (7), and (8), the corresponding algorithm (Algorithms 1, 2, 3, or 4) has fast convergence. Finding matrices P and Q depends on the matrix A. We will discuss this issue in the next section in conjunction with specific applications.

4. Applications

This section is devoted to applications of shadow block iteration methods presented in Section 2 to solving linear systems of practical importance. Our first two applications are for linear systems which often appear in signal processing (cf., [2]), one for the classical regularization and the other for the multilevel regularization. We will choose filters constructed from compactly supported wavelets to convert coefficient matrices of the systems into those having a shadow block structure. The third application is for linear systems arising from Galerkin methods for solving two-point boundary value problems of ordinary differential equations of the second order. The treatment presented here for this application is motivated by an idea initiated in [5]. For some related work, also see [3,4,10,18].

4.1. Matrices arising from the signal de-blurring problem

The classical de-blurring problem of noisy and blurred signals in the case of space invariant point spread functions is described by an integral equation. The discretization of this model by rectangle formulae leads to (see [2])

$$\mathbf{h} = F\mathbf{u} + \mathbf{w}, \tag{18}$$

where $\mathbf{h}$ is the observed signal, $\mathbf{w}$ is noise and F is a blurring matrix generated by a point spread function. The problem is then to recover the unknown object $\mathbf{u}$ when both F and $\mathbf{h}$ are available. To obtain a regularized approximation to $\mathbf{u}$ in (18), one can resort to solving the linear equation [16]

$$(\lambda I + \tilde{K})\mathbf{u} = \mathbf{g}, \tag{19}$$

where λ is an regularization parameter, $\tilde{K} = F^T F$ and $\mathbf{g} = F^T\mathbf{h}$. We will apply our proposed algorithms to solving linear systems (19). To this end, compactly supported orthogonal wavelets [6] are chosen to design transform matrices which convert the matrix $\lambda I + \tilde{K}$ into the one having a shadow block structure.

Let $\{h_\ell\colon\ \ell = 0, 1, \ldots, 2L-1\}$ and $\{g_\ell\colon\ \ell = 0, 1, \ldots, 2L-1\}$ be a pair of length $2L$ low-pass and high-pass filters associated with a compactly supported orthogonal multiresolution analysis [6]. For a given integer $n \geqslant \log_2 2L$, we define two vectors of 2^n-dimension by

$$\mathbf{s} := [h_0, \ldots, h_{2L-1}, 0, \ldots, 0]^T \quad \text{and} \quad \mathbf{t} := [g_0, \ldots, g_{2L-1}, 0, \ldots, 0]^T.$$

We define a circular shift c of any vector $\mathbf{v} = [v_1, \ldots, v_N]^T$ as

$$c(\mathbf{v}) = [v_N, v_1, \ldots, v_{N-1}]^T.$$

Associated with the low-pass filter and high-pass filter we introduce a $2^n \times 2^n$ matrix

$$P_n := \left[\mathbf{s}, c^2(\mathbf{s}), \ldots, c^{2^n-2}(\mathbf{s}), \mathbf{t}, c^2(\mathbf{t}), \ldots, c^{2^n-2}(\mathbf{t})\right]^T.$$

By using the well-known wavelet theory [6], we observe that

$$\left\langle c^\ell(\mathbf{s}), c^{\ell+2k}(\mathbf{s})\right\rangle = \delta_k, \quad \left\langle c^\ell(\mathbf{t}), c^{\ell+2k}(\mathbf{t})\right\rangle = \delta_k, \quad \left\langle c^\ell(\mathbf{s}), c^{\ell+2k}(\mathbf{t})\right\rangle = 0$$

and thus, we conclude that matrix P_n is orthogonal.

In wavelet analysis, $Q_{n,n} = P_n$ is the first level wavelet decomposition matrix. In general, the ℓth level $2^n \times 2^n$ wavelet decomposition matrix is given by

$$Q_{n,n-\ell+1} = \begin{bmatrix} P_{n-\ell+1} & \\ & I \end{bmatrix}.$$

These matrices are all orthogonal. Applying $Q_{n,n}, \ldots, Q_{n,n-\ell+1}$ to Eq. (19) yields

$$(\lambda I + K) P_{\ell,n}\mathbf{u} = P_{\ell,n}\mathbf{g}, \tag{20}$$

where

$$K = P_{\ell,n}\tilde{K} P_{\ell,n}^T$$

and

$$P_{\ell,n} := Q_{n,n-\ell+1} \cdots Q_{n,n}. \tag{21}$$

The matrix $P_{\ell,n}$ is our transformation matrix.

We perform $\ell = n - n_0$ times wavelet transforms, i.e., the transform matrix is

$$P_{n-n_0,n} = Q_{n,n_0+1} \cdots Q_{n,n}.$$

We write

$$K = \begin{bmatrix} K_1 & K_2 \\ K_3 & K_4 \end{bmatrix}$$

where K_1 is size of $2^{n_0} \times 2^{n_0}$. Correspondingly, we have that

$$A := \lambda I + K = \begin{bmatrix} \lambda I + K_1 & K_2 \\ K_3 & \lambda I + K_4 \end{bmatrix}.$$

To use our proposed algorithms, we identify

$$A_1 := \lambda I + K_1, \qquad A_2 := K_2, \qquad A_3 := K_3.$$

Appropriate choices for A_4 and A_5 are a crucial issue for convergence speed of the corresponding algorithms. The choices are based on the absolute value of the diagonal entries of matrix K. For this purpose, we define a diagonal matrix D by

$$[D]_{i,i} = \begin{cases} 0, & \text{if } 1 \leqslant i \leqslant 2^{n_0}, \\ \frac{1}{2^\ell} \sum_{k=2^\ell+1}^{2^{\ell+1}} [K]_{k,k}, & \text{if } 2^\ell + 1 \leqslant i \leqslant 2^{\ell+1} \text{ and } \ell \geqslant n_0, \end{cases}$$

and we partition matrix D according to the same partition for K, that is,

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}.$$

We suggest two choices.

Choice 1: When the diagonal entries of K are relatively small, we choose

$$A_4 = \lambda I, \qquad A_5 = K_4.$$

Choice 2: When the diagonal entries of K are relatively large, we choose

$$A_4 = \lambda I + D_2, \qquad A_5 = K_4 - D_2.$$

In the following example, we will show how these choices affect convergence speed of our algorithms. We will also discuss a motivation that leads to Choice 2.

Example 1. In this example, we choose $\lambda = 1$ and

$$F = \frac{1}{16}\begin{bmatrix} 6 & 4 & 1 & & & \\ 4 & \ddots & \ddots & \ddots & & \\ 1 & \ddots & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & 1 \\ & & \ddots & \ddots & \ddots & 4 \\ & & & 1 & 4 & 6 \end{bmatrix}_{2^n \times 2^n}.$$

In our simulation, we use the low-pass filter given by

$$\left\{\frac{1+\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{3-\sqrt{3}}{4\sqrt{2}}, \frac{1-\sqrt{3}}{4\sqrt{2}}\right\}$$

and the high-pass filter given by

$$\left\{\frac{1-\sqrt{3}}{4\sqrt{2}}, \frac{-3+\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{-1-\sqrt{3}}{4\sqrt{2}}\right\},$$

see [6]. The original "Piece–Regular" data $\mathbf{u}$ in our test are taken from the WaveLab toolbox at http://www-stat.stanford.edu/~wavelab/ developed by Donoho's research group. We use $\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$ and the root mean square error (rmse) to measure our results $\tilde{\mathbf{u}}$. We choose $n_0 = 64$ in our numerical experiments. In all numerical experiments presented in this paper, we terminate iterative algorithms when the following criterion is satisfied: $\|\mathbf{u}_{m+1} - \mathbf{u}_m\|/\|\mathbf{u}_m\| < 10^{-8}$.

Table 1 lists the numerical results of four algorithms using Choice 1. It can be seen that the numbers of iteration (Ite) increase dramatically with the size of matrix K. This is because the diagonal elements $[K]_{i,i}$, $i > 2^{n_0}$ are still very large, which can be seen from the left of Fig. 2, and thus, it prevents our proposed algorithms working efficiently. Hence, this choice gives slow convergence. The plots of diagonal elements $K - D$ and $I + D$ are shown in Fig. 2 (middle and right). The idea of Choice 2 is to adjust the

Table 1
The numerical results for Example 1 with Choice 1 and $\lambda = 1$

n	Algorithm 1			Algorithm 2		
	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse
7	11	7.469e-8	1.013e-8	8	6.058e-8	7.378e-9
8	27	2.674e-7	5.368e-8	26	2.113e-7	5.547e-8
9	104	2.338e-7	7.706e-8	104	2.215e-7	7.502e-8
10	420	2.522e-7	8.674e-8	420	2.491e-7	8.625e-8
11	1681	2.651e-7	8.849e-8	168	2.642e-7	8.835e-8
n	Algorithm 3			Algorithm 4		
	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse
7	7	3.821e-8	6.173e-9	5	1.128e-08	1.446e-9
8	25	1.216e-7	5.680e-8	13	8.506e-8	3.344e-8
9	104	2.173e-7	7.400e-8	52	2.222e-7	7.486e-8
10	420	2.480e-7	8.629e-8	210	2.494e-7	8.654e-8
11	1681	2.641e-7	8.833e-8	841	2.617e-7	8.754e-8

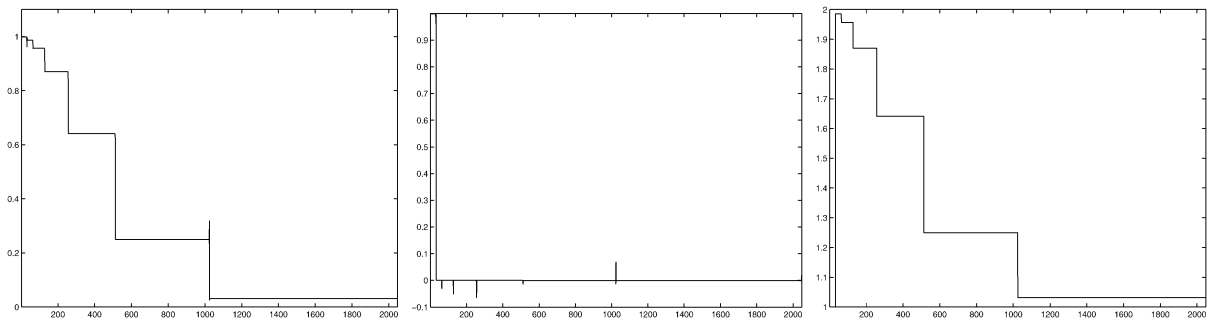

Fig. 2. Plots (from left to right) of diagonal elements of matrices K, $K - D$, and $I + D$, respectively.

Table 2
The numerical results for Example 1 with Choice 2 and $\lambda = 1$

n	Algorithm 1			Algorithm 2		
	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse
7	10	1.045e-7	1.503e-8	7	2.298e-8	2.855e-9
8	14	8.635e-8	1.292e-8	13	6.866e-8	1.348e-8
9	13	2.774e-7	2.575e-8	13	2.149e-7	1.823e-8
10	13	2.498e-7	1.542e-8	13	2.050e-7	1.247e-8
11	13	2.408e-7	1.030e-8	12	7.520e-7	3.399e-8
12	12	7.241e-7	2.724e-8	12	7.277e-7	2.485e-8
n	Algorithm 3			Algorithm 4		
	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse	Ite	$\|\mathbf{u} - \tilde{\mathbf{u}}\|_\infty$	rmse
7	6	2.434e-8	3.818e-9	5	3.901e-9	5.157e-10
8	12	6.682e-8	2.502e-8	6	1.299e-7	3.292e-8
9	12	1.762e-7	3.021e-8	6	1.762e-7	3.128e-8
10	12	1.051e-7	1.704e-8	6	1.221e-7	1.792e-8
11	12	1.108e-7	1.148e-8	6	1.373e-7	1.208e-8
12	11	4.585e-7	3.472e-8	6	1.659e-7	8.709e-9

matrices I and K accordingly. The numerical results with Choice 2 are given in Table 2. They show that our proposed algorithms work well with this choice in this example.

We denote by E_1, E_2, E_3, and E_4 the left-hand sides of inequalities (9), (11), (13), and (14), respectively. Table 3 shows the values of E_i, the norm $\|B^{-1}C\|$, and the spectral radius $\rho(B^{-1}C)$ for our proposed algorithms with Choice 2. These numerical results confirm the theoretical results of the first four propositions presented in Section 3.

4.2. Matrices coming from multilevel regularization method for signal de-blurring problem

We again consider the signal de-blurring model (18). This time, however, instead of solving the linear equation (19) with a *single* regularization parameter λ, we consider a regularized equation with *multiple* regularization parameters. Specifically, we consider equation

$$(\Lambda + K) P_{\ell,n} \mathbf{u}_\Lambda = P_{\ell,n} \mathbf{g}, \tag{22}$$

Table 3
The values of $\rho(B^{-1}C)$, $\|B^{-1}C\|$, and E_i for Example 1 with Choice 2

		$n=7$	$n=8$	$n=9$	$n=10$
Alg. 1	E_1	0.176409	0.364336	0.352662	0.344825
	$\|B^{-1}C\|$	0.162085	0.274323	0.269848	0.269906
	$\rho(B^{-1}C)$	0.149872	0.271663	0.265638	0.264798
Alg. 2	E_2	0.177440	0.393709	0.408606	0.410251
	$\|B^{-1}C\|$	0.158988	0.265945	0.269561	0.269987
	$\rho(B^{-1}C)$	0.041706	0.249399	0.261676	0.263271
Alg. 3	E_3	0.115980	0.289502	0.290529	0.288423
	$\|B^{-1}C\|$	0.112546	0.253607	0.265868	0.267338
	$\rho(B^{-1}C)$	0.041706	0.249399	0.261676	0.263271
Alg. 4	E_4	0.020460	0.096599	0.097114	0.095608
	$\|B^{-1}C\|$	0.017634	0.065964	0.069824	0.070066
	$\rho(B^{-1}C)$	0.017493	0.065667	0.069490	0.069744

where $P_{\ell,n}$ is the transformation matrix defined by (21),

$$K := P_{\ell,n}\tilde{K}P_{\ell,n}^T,$$

and $(\ell+1)$-level regularization matrix

$$\Lambda := \begin{bmatrix} \lambda_1 I_{2^{n_0}} & & & & \\ & \lambda_2 I_{2^{n_0}} & & & \\ & & \lambda_3 I_{2^{n_0+1}} & & \\ & & & \ddots & \\ & & & & \lambda_{\ell+1} I_{2^{n-1}} \end{bmatrix} \tag{23}$$

with positive λ_i, $i = 1, 2, \ldots, \ell+1$. Intuitively, the matrix $P_{\ell,n}\tilde{K}P_{\ell,n}^T$ has a multilevel "shadow" structure. This motivates the introduction of the matrix Λ with multiple regularization parameters in order to tailor to the multilevel structure of the underlying matrix. For more information regarding the multiple parameter regularization, see [11] and [15].

We reformulate Eq. (22) into a convenient form. To this end, we partition matrix $P_{\ell,n}$ as

$$P_{\ell,n} = \left(T_1^T, T_2^T, \ldots, T_{l+1}^T\right)^T,$$

where T_1 is a $2^{n_0} \times 2^n$ matrix and T_i is a $2^{n_0+i-2} \times 2^n$ matrix, for $i \geqslant 2$. Noticing that

$$P_{\ell,n}\left(\sum_{i=1}^{\ell+1} \lambda_i T_i^T T_i\right) P_{\ell,n}^T = \Lambda$$

and $P_{\ell,n}^T K P_{\ell,n} = \tilde{K} = F^T F$, we rewrite Eq. (22) into an equivalent form

$$\left(F^T F + \sum_{i=1}^{\ell+1} \lambda_i T_i^T T_i\right)\mathbf{u} = F^T \mathbf{h}. \tag{24}$$

We remark that solving (24) is equivalent to minimizing a cost functional with wavelet regularization terms as follows

$$\min_{\mathbf{u}}\left\{\|F\mathbf{u}-\mathbf{h}\|^2+\sum_{i=1}^{\ell+1}\lambda_i\|T_i\mathbf{u}\|^2\right\}. \tag{25}$$

The crucial issue for the regularization method is the choice of regularization parameters. Many approaches have been proposed in the literature. Among them, discrepancy principle method [14], generalized cross-validation method [7], L-curve criterion method [9], have proved to be useful in numerous applications. We consider three different strategies for the choice of regularization parameters, based on different knowledge of the linear system.

Strategy 1. Suppose that we have a prior knowledge on noise. In uniform parameter situation, a natural choice of regularization parameter is $\|\mathbf{w}\|^2/\|\mathbf{u}\|^2$, see [13]. We extend this idea to the multiple parameter case and choose the parameter at each level as

$$\lambda_i=\left(\frac{\|T_i\mathbf{w}\|}{\|T_i\mathbf{h}\|}\right)^2,\quad i=1,2,\ldots,\ell+1,$$

where we use $\|T_i\mathbf{h}\|$ as a good approximation of $\|T_i\mathbf{u}\|$. Our experiments show that it is a feasible choice.

Strategy 2. We observe that K has a multilevel structure

$$K=\left(K_1^T,K_2^T,\ldots,K_{\ell+1}^T\right)^T,$$

where K_1 is of size $2^{n_0}\times 2^n$ and K_i is of size $2^{n_0+i-2}\times 2^n$ for $i\geqslant 2$. We compute the singular values of K_i^T and their mean σ_i. To balance the distribution of the singular values along the levels, we set

$$\sigma_{max}:=\max\{\sigma_i\colon\ i=1,2,\ldots,\ell+1\},$$

and choose

$$\lambda_i=\sigma_{\max}-\sigma_i,\quad \text{for } i=1,2,\ldots,\ell+1.$$

Strategy 3. If we have no prior knowledge on noise, we adjust the diagonal elements of K according to the multilevel structure. We define the mean of diagonal element at each level by

$$d_i=\begin{cases}\frac{1}{2^{n_0}}\sum_{k=1}^{2^{n_0}}[K]_{k,k}, & \text{if } i=1,\\ \frac{1}{2^{n_0-2+i}}\sum_{k=2^{n_0-2+i}+1}^{2^{n_0-1+i}}[K]_{k,k}, & \text{if } i=2,3,\ldots,\ell+1.\end{cases}$$

Our idea is to balance the mean of diagonal elements at each level. We define

$$d_{\max}:=\max\{d_i\colon\ i=1,2,\ldots,\ell+1\}$$

and choose

$$\lambda_i=d_{\max}-d_i,\quad \text{for } i=1,2,\ldots,\ell+1.$$

Now suppose that the matrix Λ has been chosen according to one of the strategies and we consider the numerical solution of the corresponding linear system. To apply our iteration algorithms, we write

$$\Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \quad \text{with } \Lambda_1 = \lambda_1 I_{2^{n_0}},$$

and correspondingly, we have that

$$A := \Lambda + K = \begin{bmatrix} \Lambda_1 + K_1 & K_2 \\ K_3 & \Lambda_2 + K_4 \end{bmatrix}.$$

As in the last subsection, we choose

$$A_1 = \Lambda_1 + K_1, \qquad A_2 = K_2, \qquad A_3 = K_3,$$

and again suggest two choices for A_4 and A_5.

Choice 1. When the diagonal entries of K are relatively small, we choose

$$A_4 = \Lambda_2, \qquad A_5 = K_4.$$

Choice 2. When the diagonal entries of K are relatively large, we choose

$$A_4 = \Lambda_2 + D_2, \qquad A_5 = K_4 - D_2.$$

In the next example, we demonstrate the use of the shadow block iterated method for solving the linear system arising from this context. We will also compare the effect of three strategies for the choice of parameters.

Example 2. We inherit F and **u** from Example 1. Fix the length of data as 512 and add the Gaussian noise with zero mean and variance $\sigma = 2.5$ to **h**. Denote

$$\mathbf{g} = F^T \mathbf{h}.$$

Use the same low-pass filter and high-pass filter as those used in Example 1. We generate the transform matrix by using the compactly supported orthogonal wavelet and apply it to $\tilde{K}$. We perform the wavelet transform three times and decompose **g** into four levels. Four regularization parameters λ_i, $i = 1, 2, 3, 4$ are chosen for each level, according to the three strategies described earlier. The values of these parameters are shown in Table 4. In this table and what follows, we use the abbreviation "Str." for "Strategy."

To apply our iteration algorithms, we write Λ and K as

$$\Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}, \qquad K = \begin{bmatrix} K_1 & K_2 \\ K_3 & K_4 \end{bmatrix},$$

Table 4
Multilevel regularization parameters with three different strategies

	λ_1	λ_2	λ_3	λ_4
MRM (Str. 1)	0.003803	0.104236	0.521359	0.892243
MRM (Str. 2)	0	0.263502	0.642263	0.846484
MRM (Str. 3)	0	0.279180	0.670181	0.888467

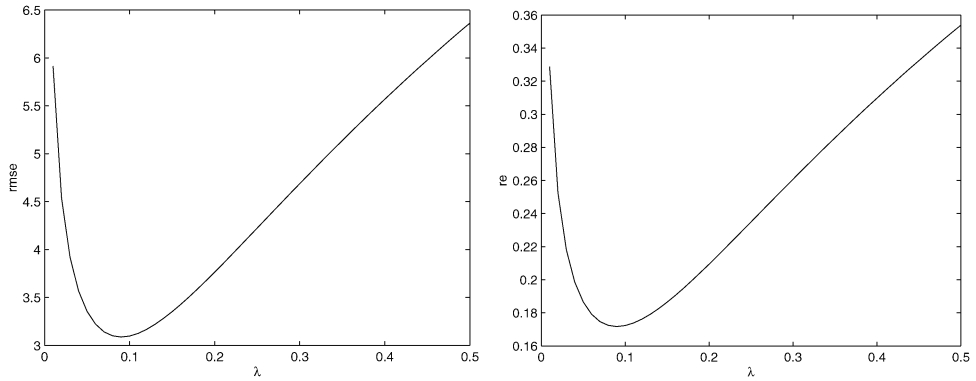

Fig. 3. Values of rmse and re of the recovered signal with a uniform regularization parameter for the signal corrupted by Gaussian white noise with zero mean and variance $\sigma = 2.5$.

where Λ_1, K_1 are of size 64×64. Accordingly, we have the partition

$$A := \Lambda + K = \begin{bmatrix} \Lambda_1 + K_1 & K_2 \\ K_3 & \Lambda_2 + K_4 \end{bmatrix}$$

and we choose

$$A_1 := \Lambda_1 + K_1, \qquad A_2 := K_2, \qquad A_3 := K_3, \qquad A_4 := \Lambda_2 + D_2, \qquad A_5 := K_4 - D_2.$$

We present the ability of uniform regularization method in Fig. 3. Root mean square error (rmse) and relative error (re) are used to measure the errors, where re is defined as $\|\mathbf{u}_\Lambda - \mathbf{u}\|/\|\mathbf{u}\|$. Uniform regularization parameter varies from 0.01 to 0.5 with step 0.01 and the recovered result is obtained by choosing the best uniform regularization parameter. Our experiments show that $\lambda = 0.08$ is near-optimal in the terms of values of rmse $= 3.303834$ and re $= 0.183727$. The improvement of using multilevel regularization method (MRM) over the one using uniform regularization method are shown in Table 5, Table 6, and Fig. 4 for four iteration algorithms with different strategies. The stopping criterion is $\|\mathbf{u}_{m+1} - \mathbf{u}_m\|/\|\mathbf{u}_m\| < 10^{-8}$. The values of E_i, the norm $\|B^{-1}C\|$ and the spectral radius $\rho(B^{-1}C)$ are provided in Table 7.

Table 5
The number of iterations for Example 2 with Choice 2

	Ite (Algorithm 1)	Ite (Algorithm 2)	Ite (Algorithm 3)	Ite (Algorithm 4)
MRM (Str. 1)	18	18	18	10
MRM (Str. 2)	17	17	16	9
MRM (Str. 3)	16	15	15	9

Table 6
The numerical result for Example 2 with Choice 2

	MRM (Str. 1)	MRM (Str. 2)	MRM (Str. 3)
rmse	2.324671	2.334616	2.341564
re	0.129275	0.129828	0.130215

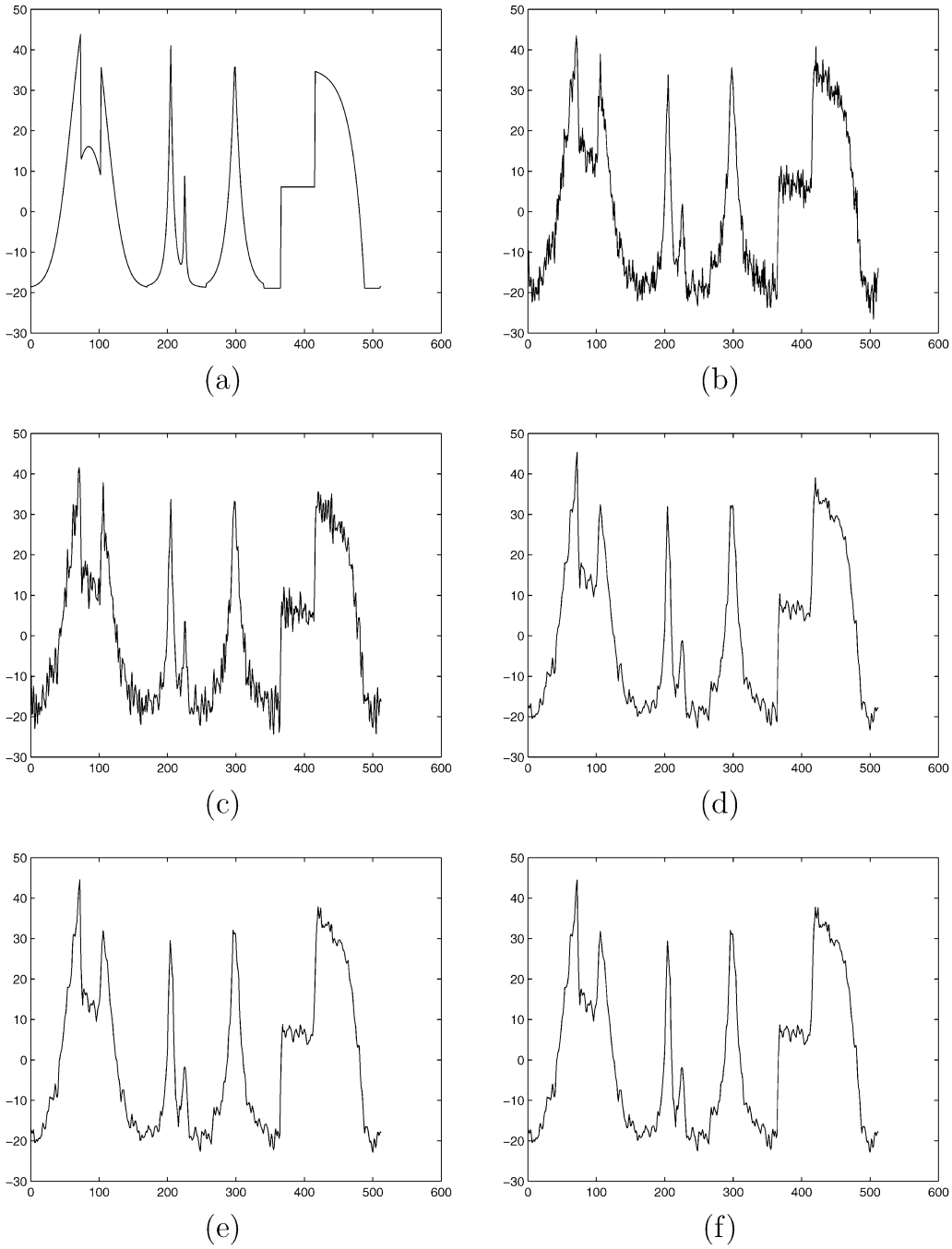

Fig. 4. (a) The original signal; (b) The blurred and noisy data; (c) The restored signal by uniform regularization method with almost least rmse; (d) The restored signal by MRM with strategy 1; (e) The restored signal by MRM with strategy 2; (f) The restored signal by MRM with strategy 3.

Table 7
The values of $\rho(B^{-1}C)$, $\|B^{-1}C\|$, and E_i for Example 2 with four different methods

		MRM (Str.1)	MRM (Str.2)	MRM (Str.3)
Algorithm 1	E_1	0.349359	0.293744	0.281183
	$\|B^{-1}C\|$	0.398194	0.366403	0.359832
	$\rho(B^{-1}C)$	0.396180	0.360771	0.350986
Algorithm 2	E_2	0.311047	0.225277	0.205771
	$\|B^{-1}C\|$	0.391777	0.350239	0.338662
	$\rho(B^{-1}C)$	0.381298	0.338635	0.327601
Algorithm 3	E_3	0.238073	0.207583	0.200226
	$\|B^{-1}C\|$	0.392428	0.362864	0.358921
	$\rho(B^{-1}C)$	0.381298	0.338635	0.327601
Algorithm 4	E_4	0.061279	0.041629	0.037698
	$\|B^{-1}C\|$	0.150122	0.122053	0.115821
	$\rho(B^{-1}C)$	0.148399	0.120276	0.113868

4.3. Matrices obtained from Galerkin methods for two-point boundary value problems

We show in this section the use of our proposed algorithms in solving linear systems resulting from the two-point Dirichlet boundary value problems of ordinary differential equations of the second order. Specifically, we consider the boundary value problem

$$\begin{cases} -u''(t) + \alpha_1(t)u'(t) + \alpha_2(t)u(t) = f(t), \quad t \in (0,1), \\ u(0) = u(1) = 0, \end{cases} \tag{26}$$

where $\alpha_1, \alpha_2 \in L^\infty([0,1])$, and $f \in L^2([0,1])$. Problem (26) has a unique solution u. We next describe the Galerkin method for solving the boundary value problem using linear splines.

For this purpose, we let ϕ be the piecewise linear function on the interval $[0,1]$, i.e.,

$$\phi(t) = \begin{cases} 2t, & 0 \leqslant t < 1/2, \\ 2-2t, & 1/2 \leqslant t \leqslant 1. \end{cases}$$

Set

$$\phi_{n,k}(t) := \phi\big(2^{n-1}t - k/2\big), \quad t \in [0,1], \text{ for } k = 0, 1, \ldots, 2^n - 2, \ n \geqslant 1$$

and define a (2^n-1)-dimensional space by

$$\mathcal{V}_n := \operatorname{span}\big\{\phi_{n,k}\colon\ k = 0, 1, \ldots, 2^n - 2\big\}.$$

For any function $\tilde{u} \in \mathcal{V}_n$, one has

$$\tilde{u}(t) = \sum_{k=0}^{2^n-2} u_{k+1}\phi_{n,k}(t), \quad t \in [0,1]. \tag{27}$$

Clearly, $u_k = \tilde{u}(k/2^n)$ for $k = 0, 1, \ldots, 2^n - 2$. We say that $\tilde{u} \in \mathcal{V}_n$ is an Galerkin approximation of the exact solution u of (26) if $\tilde{u}$ satisfies the following equation:

$$\langle \tilde{u}', \phi'_{n,i}\rangle + \langle \alpha_1\tilde{u}', \phi_{n,i}\rangle + \langle \alpha_2\tilde{u}, \phi_{n,i}\rangle = \langle f, \phi_{n,i}\rangle, \quad i = 0, 1, \ldots, 2^n - 2.$$

By setting $\mathbf{u} := [u_1, u_2, \ldots, u_{2^n-1}]^T$, we see that the above equation is equivalent to the linear system

$$(J_n + F_n + G_n)\mathbf{u} = \mathbf{g}, \tag{28}$$

where

$$[J_n]_{i,j} = 2^{-n-1}\langle \phi'_{n,j}, \phi'_{n,i}\rangle, \qquad [F_n]_{i,j} = 2^{-n-1}\langle \alpha_1 \phi'_{n,j}, \phi_{n,i}\rangle,$$
$$[G_n]_{i,j} = 2^{-n-1}\langle \alpha_2 \phi_{n,j}, \phi_{n,i}\rangle, \qquad [\mathbf{g}]_i = 2^{-(n+1)/2}\langle f, \phi_{n,i}\rangle.$$

To solve the linear system (28), we first study useful properties of matrix J_n. A direct computation gives

$$J_n = \frac{1}{2}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \tag{29}$$

which is a $(2^n - 1) \times (2^n - 1)$ symmetric Toeplitz matrix and it is independent of functions α_1 and α_2.

We next demonstrate how to obtain from this matrix J_n the matrix having a *shadow* block structure by two matrices (transforms) P and Q. To this end, we define the shift σ of any vector $\mathbf{v} = [v_1, v_2, \ldots, v_{2^n-1}]^T$ by setting

$$\sigma(\mathbf{v}) := [0, v_1, \ldots, v_{2^n-2}]^T.$$

To define our transforms, we need two $(2^n - 1)$-dimensional vectors

$$\mathbf{p}_n := \begin{bmatrix} \frac{\sqrt{2}}{2} & \sqrt{2} & \frac{\sqrt{2}}{2} & 0 & \ldots & 0 \end{bmatrix}^T \quad \text{and} \quad \mathbf{e}_n := [\,1 \quad 0 \quad \ldots \quad 0\,]^T.$$

Associated with the shift operator σ and vectors $\mathbf{p}_n$ and $\mathbf{e}_n$, we define a matrix

$$P_n = \big[\, \mathbf{p}_n \quad \sigma^2(\mathbf{p}_n) \quad \ldots \quad \sigma^{2^n-4}(\mathbf{p}_n) \quad \mathbf{e}_n \quad \sigma^2(\mathbf{e}_n) \quad \ldots \quad \sigma^{2^n-2}(\mathbf{e}_n) \,\big]^T.$$

For any integers n and $2 \leqslant \ell \leqslant n$, we define the transform by the $(2^n - 1) \times (2^n - 1)$ matrix

$$Q_{n,\ell} := \begin{bmatrix} P_\ell & \\ & I \end{bmatrix}$$

and note that $Q_{n,n} = P_n$.

The following lemma shows how a shadow block of J_n is obtained by applying the transform $Q_{n,n}$.

Lemma 2. *If $n \geqslant 2$ is any positive integer, then*

$$Q_{n,n} J_n Q_{n,n}^T = \begin{bmatrix} J_{n-1} & \\ & I \end{bmatrix}. \tag{30}$$

Proof. Note that the matrix J_n can be written as

$$J_n = D_n D_n^T$$

where $(2^n-1)\times 2^n$ matrix D_n is a *difference* matrix, i.e.,

$$D_n=\frac{\sqrt{2}}{2}\begin{bmatrix}1 & -1 & & & & \\ & 1 & -1 & & & \\ & & \ddots & \ddots & & \\ & & & 1 & -1 & \\ & & & & 1 & -1\end{bmatrix}.$$

We also define two vectors of 2^n dimension

$$\mathbf{s}_n=\Big[\frac{1}{2},\frac{1}{2},-\frac{1}{2},-\frac{1}{2},\underbrace{0,\ldots,0}_{2^n-4}\Big]^T \quad\text{and}\quad \mathbf{t}_n=\Big[\frac{\sqrt{2}}{2},-\frac{\sqrt{2}}{2},\underbrace{0,\ldots,0}_{2^n-2}\Big]^T.$$

In terms of $\mathbf{s}_n$ and $\mathbf{t}_n$, we have

$$Q_{n,n}D_n=\big[\mathbf{s}_n \quad \sigma^2(\mathbf{s}_n) \quad \ldots \quad \sigma^{2^n-4}(\mathbf{s}_n) \quad \mathbf{t}_n \quad \sigma^2(\mathbf{t}_n) \quad \ldots \quad \sigma^{2^n-2}(\mathbf{t}_n)\big]^T.$$

It is easy to verify that

$$\big(\sigma^{2k}(\mathbf{s}_n)\big)^T\sigma^{2\ell}(\mathbf{s}_n)=\begin{cases}1, & \text{if } k=\ell,\\ -\frac{1}{2}, & \text{if } |k-\ell|=1,\\ 0, & \text{if } |k-\ell|\geqslant 2,\end{cases}$$

$$\big(\sigma^{2k}(\mathbf{t}_n)\big)^T\sigma^{2\ell}(\mathbf{t}_n)=\begin{cases}1, & \text{if } |k-\ell|=0,\\ 0, & \text{if } |k-\ell|\geqslant 1,\end{cases}$$

for all possible k and ℓ, and

$$\big(\sigma^{2k}(\mathbf{s}_n)\big)^T\sigma^{2\ell}(\mathbf{t}_n)=0, \quad \text{for } 0\leqslant k\leqslant 2^{n-1}-2 \text{ and } 0\leqslant \ell\leqslant 2^{n-1}-1.$$

Employing these formulas and the equation

$$Q_{n,n}J_nQ_{n,n}^T=(Q_{n,n}D_n)(Q_{n,n}D_n)^T,$$

we have the desired result (30). □

For matrix J_n, Lemma 2 says that the exact "shadow" J_{n-1} of J_n can be obtained from the transform matrix $Q_{n,n}$. In general, we have the following result.

Proposition 9. *If $n\geqslant 2$ is any integer, then*

$$Q_{n,2}\cdots Q_{n,n}J_nQ_{n,n}^T\cdots Q_{n,2}^T=I. \tag{31}$$

Proof. Applying (30) recursively, we have that

$$Q_{n,\ell}\cdots Q_{n,n}J_nQ_{n,n}^T\cdots Q_{n,\ell}^T=\begin{bmatrix}J_{\ell-1} & \\ & I\end{bmatrix} \tag{32}$$

for all $2\leqslant\ell\leqslant n$. In particular, when $\ell=2$ we obtain the result of this theorem by noticing that $J_1=[1]$. □

Proposition 9 suggests that it is convenient to choose

$$P=Q=Q_{n,2}\cdots Q_{n,n}.$$

Note that the choice of P and Q in this case is independent of functions α_1, α_2. Hence, the proposed algorithms are applicable to a large class of problems since they do not depend on the other two matrices F_n, G_n. We next illustrate the methods by a specific numerical example.

Example 3. In (26), we choose

$$\alpha_1(t) = t, \qquad \alpha_2(t) = -\pi^2, \quad f(t) = \pi t \cos(\pi t).$$

The exact solution of (26) is then given by

$$u(t) := \sin(\pi t), \quad t \in [0, 1].$$

Correspondingly, for any integer n, both F_n and G_n are tri-diagonal matrices with

$$[F_n]_{i,i+1} = \frac{1+3i}{12\cdot 4^n}, \qquad [F_n]_{i+1,i} = -\frac{2+3i}{12\cdot 4^n}, \qquad [F_n]_{i,i} = -\frac{1}{6\cdot 4^n}$$

and

$$[G_n]_{i,i+1} = -\frac{\pi^2}{12\cdot 4^n}, \qquad [G_n]_{i+1,i} = -\frac{\pi^2}{12\cdot 4^n}, \qquad [G_n]_{i,i} = -\frac{\pi^2}{3\cdot 4^n}.$$

For a continuous function f, we define the backward difference operator Δ by

$$(\Delta f)\left(\frac{i}{2^n}\right) := f\left(\frac{i}{2^n}\right) - f\left(\frac{i-1}{2^n}\right).$$

We also let

$$f_1(t) = \frac{t}{\pi}\sin\pi t + \frac{1}{\pi^2}\cos\pi t \quad \text{and} \quad f_2(t) = \frac{t^2}{\pi}\sin\pi t + \frac{2t}{\pi^2}\cos\pi t - \frac{2}{\pi^3}\sin\pi t, \quad t \in [0, 1]$$

and define vector $\mathbf{g}$ by

$$[\mathbf{g}]_i = 2^{(n-1)/2}\pi\left[-\left(\Delta^2 f_2\right)\left(\frac{i+1}{2^n}\right) - \frac{i-1}{2^n}(\Delta f_1)\left(\frac{i}{2^n}\right) + \frac{i+1}{2^n}(\Delta f_1)\left(\frac{i+1}{2^n}\right)\right].$$

By Proposition 9, Eq. (28) is transformed to

$$(I + K)\tilde{\mathbf{u}} = Q_{n,2}\cdots Q_{n,n}\mathbf{g},$$

where

$$\tilde{\mathbf{u}} = \left(Q_{n,n}^T\cdots Q_{n,2}^T\right)^{-1}\mathbf{u}$$

and

$$K = Q_{n,2}\cdots Q_{n,n}(F_n + G_n)Q_{n,n}^T\cdots Q_{n,2}^T.$$

Figure 5(a) shows the values of the entries of matrix $F_6 + G_6$ while Fig. 5(b) shows those of matrix $Q_{6,2}\cdots Q_{6,6}(F_6 + G_6)Q_{6,6}^T\cdots Q_{6,2}^T$.

To apply the four proposed algorithms to this example, we write K as

$$K = \begin{bmatrix} K_1 & K_2 \\ K_3 & K_4 \end{bmatrix},$$

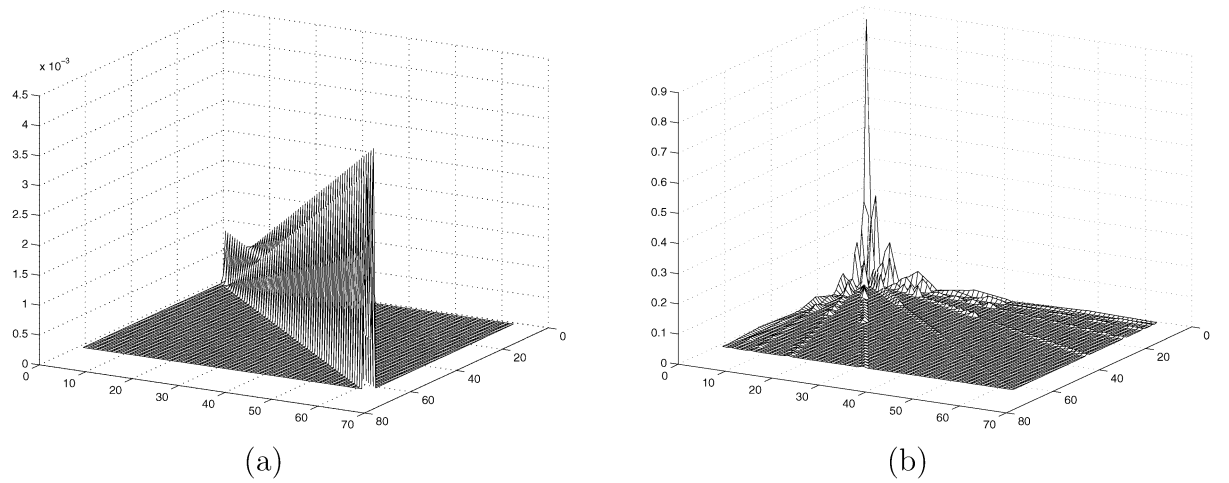

Fig. 5. The definition of F_6, G_6, and $Q_{6,\ell}$ are given in Example 3. (a) The absolute values of the matrix $F_6 + G_6$; (b) The absolute values of the matrix $Q_{6,2} \cdots Q_{6,6}(F_6 + G_6)Q_{6,6}^T \cdots Q_{6,2}^T$.

Table 8
The numerical results with varying sizes of A

n	Order of A (2^n)	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4	
		Ite	err	Ite	err	Ite	err	Ite	err
7	128	8	0.0012	5	0.0012	5	0.0012	4	0.0012
8	256	8	2.9995e-4	5	2.9995e-4	5	2.9995e-4	4	2.9995e-4
9	512	8	7.4973e-5	5	7.4973e-5	5	7.4973e-5	4	7.4973e-5
10	1024	8	1.8742e-5	5	1.8743e-5	5	1.8743e-5	4	1.8742e-5
11	2048	8	4.6864e-6	5	4.6870e-6	5	4.6870e-6	4	4.6864e-6
12	4096	8	1.1777e-6	5	1.1782e-6	5	1.1782e-6	4	1.1777e-6
13	8192	8	3.1437e-7	5	3.1492e-7	5	3.1492e-7	4	3.1437e-7

where K_1 is of order 63×63. Accordingly, we have that

$$A := I + K = \begin{bmatrix} I + K_1 & K_2 \\ K_3 & I + K_4 \end{bmatrix}.$$

Choosing

$$A_1 := I + K_1, \qquad A_2 := K_2, \qquad A_3 := K_3, \qquad A_4 := I, \qquad A_5 := K_4,$$

we obtain the three splittings (B, C) of A as described in (6), (7), and (8).

The results of the maximum number of iterates (Ite) and the maximum error $\text{err} = \max_i |\sin(\pi i/2^n) - [\mathbf{u}_m]_i|$ are given in Table 8. We remark that the larger n is, the smaller max-error err is. Moreover, we observe that the number of iteration for the convergence of each algorithm is independent of the size of signals.

The numerical results show that all four algorithms converge very fast and it seems that Algorithm 4 converges faster than any of other algorithms for this example. The values of E_i, the norm $\|B^{-1}C\|$, and the spectral radius $\rho(B^{-1}C)$ are given in Table 9. We can see that the value of E_i is very close to the

Table 9
The values of $\rho(B^{-1}C)$, $\|B^{-1}C\|$, and E_i for Example 3 with four different methods

		$n=7$	$n=8$	$n=9$	$n=10$	$n=11$	$n=12$
Algorithm 1	E_1	0.013211	0.015761	0.016354	0.010215	0.007180	0.007032
	$\|B^{-1}C\|$	0.013204	0.014799	0.015172	0.008933	0.005893	0.005795
	$\rho(B^{-1}C)$	0.012616	0.014078	0.014421	0.007688	0.004621	0.004916
Algorithm 2	E_2	0.013212	0.015633	0.016178	0.009903	0.006722	0.006562
	$\|B^{-1}C\|$	0.013108	0.014693	0.015063	0.008838	0.005888	0.005729
	$\rho(B^{-1}C)$	0.000367	0.001939	0.002340	0.002436	0.002460	0.002465
Algorithm 3	E_3	0.013110	0.014790	0.015205	0.009079	0.006117	0.005974
	$\|B^{-1}C\|$	0.013110	0.014685	0.015053	0.008808	0.005846	0.005716
	$\rho(B^{-1}C)$	0.000367	0.001939	0.002340	0.002436	0.002460	0.002465
Algorithm 4	E_4	0.000173	0.000231	0.000245	0.000089	0.000041	0.000039
	$\|B^{-1}C\|$	0.000163	0.000205	0.000216	0.000074	0.000030	0.000031
	$\rho(B^{-1}C)$	0.000156	0.000194	0.000204	0.000057	0.000021	0.000023

corresponding norm $\|B^{-1}C\|$ and the transformation matrix produces small spectral radius $\rho(B^{-1}C)$ for varying numbers n.

5. Conclusion

The shadow block iteration method proposed in this paper aims at efficient fast solutions of linear systems resulting from wavelet transforms. Coefficient matrices of linear systems of this type have a shadow block structure. We make use of this special structure in designing the algorithms. Theoretical and numerical study of these algorithms have confirmed that the algorithms are very efficient for solving linear systems of this type.

References

[1] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H.V. der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, second ed., SIAM, Philadelphia, PA, 1994.
[2] M. Bertero, P. Boccacci, Introduction to Inverse Problems in Imaging, Institute of Physics Publ., Bristol, 1998.
[3] Z. Chen, C.A. Micchelli, Y. Xu, Fast collocation method for second kind integral equations, SIAM J. Numer. Anal. 40 (2002) 344–375.
[4] Z. Chen, B. Wu, Y. Xu, Multilevel augmentation methods for solving operator equations, Numer. Math. J. Chinese Univ., in press.
[5] Z. Chen, B. Wu, Y. Xu, Multilevel augmentation methods for differential equations, Adv. Comput. Math., in press.
[6] I. Daubechies, Ten Lectures on Wavelets, in: CBMS Conference Series in Applied Mathematics, vol. 61, SIAM, Philadelphia, 1992.
[7] G. Golub, M. Heath, G. Wahba, Generalized cross-validation as a method for choosing a good ridge parameter, Technometrics 21 (1979) 215–223.
[8] G. Golub, C. Van Loan, Matrix Computations, second ed., Johns Hopkins, 1989.
[9] P. Hansen, Analysis of discrete ill-posed problems by means of the L-curve, SIAM Rev. 34 (1992) 561–580.

[10] R.-Q. Jia, S. Liu, Wavelet bases of Hermite cubic splines on intervals, Adv. Comput. Math., in press.
[11] Y. Lu, Y. Xu, Multiple parameter regularization for block matrices, Preprint, 2004.
[12] S. Mallat, A Wavelet Tour of Signal Processing, second ed., Academic Press, 1999.
[13] K. Miller, Least squares methods for ill-posed problems with a prescribed bound, SIAM J. Math. Anal. 1 (1970) 52–74.
[14] V. Morozov, Methods for Solving Incorrectly Posed Problems, Springer-Verlag, 1984.
[15] L. Shen, Image deconvolution using wavelet regularization, J. Electron. Imaging 11 (2002) 5–11.
[16] A. Tikhonov, V. Arsenin, Solutions to Ill-Posed Problems, Wiley, New York, 1977.
[17] R. Varga, Matrix Iterative Analysis, second ed., revised and expanded, Prentice–Hall, Englewood Cliffs, NJ, 1962, Springer, Berlin/Heidelberg/New York, 2000.
[18] J. Xu, W.C. Shann, Galerkin-wavelet methods for two-point boundary value problems, Numer. Math. 63 (1992) 123–144.